# Hackathon Veolia

## Problem:

Veolia, a multinational with operations in water, waste, and energy management, is committed to ecological transformation to improve our world. However, we face significant challenges due to the complexity of integrating technology and digitization into our processes. Currently, the company uses a variety of sensors and data sources that generate a considerable volume of real-time data. The diversity of sensors, brands, and communication protocols complicates the management, configuration, and validation of this data, which is crucial for the efficiency of our processes. To date, Veolia lacks a platform that properly manages the configuration of sensors, the format of the data, the validation of its quality, and its correct integration into our Data Lake hosted on Google Cloud Platform (GCP). This lack of appropriate infrastructure results in delays in data analysis, frequent reconfiguration needs, and an excessive dependency on external providers to perform these essential tasks. Innovation Challenge: At Veolia, we are facing the challenge of optimizing the management and analysis of data from a wide range of sensors used in our water, waste, and energy management operations. To overcome this obstacle, we aim to develop a comprehensive platform that not only connects and manages these on-site sensors but also efficiently handles real-time data transmission across various communication protocols. The goal is to create a robust solution that allows for data quality validation and facilitates its seamless integration into our Data Lake on Google Cloud Platform (GCP). This platform should offer an intuitive and user-friendly frontend, along with a powerful backend to ensure efficiency and security in data processing and storage. Furthermore, the solution must be scalable and adaptable, capable of incorporating new sensors and data sources as our needs and technologies evolve.

## Innovation Problem:

Veolia, a multinational company operating in water, waste, and energy management, is committed to ecological transformation to improve our world. However, we face significant challenges due to the complexity of integrating technology and digitization into our processes. Currently, the company uses a variety of sensors and information sources that generate a considerable volume of real-time data. The diversity of sensors, brands, and communication protocols complicates the administration, configuration, and validation of this data—crucial aspects for the efficiency of our processes.

As of today, Veolia lacks a platform capable of properly managing sensor configuration, data formatting, data quality validation, and seamless integration into our Data Lake, hosted on Google Cloud Platform (GCP). This lack of adequate infrastructure results in delays in data

analysis, frequent reconfiguration needs, and an excessive reliance on external providers to carry out these essential tasks.

# Innovation Challenge:

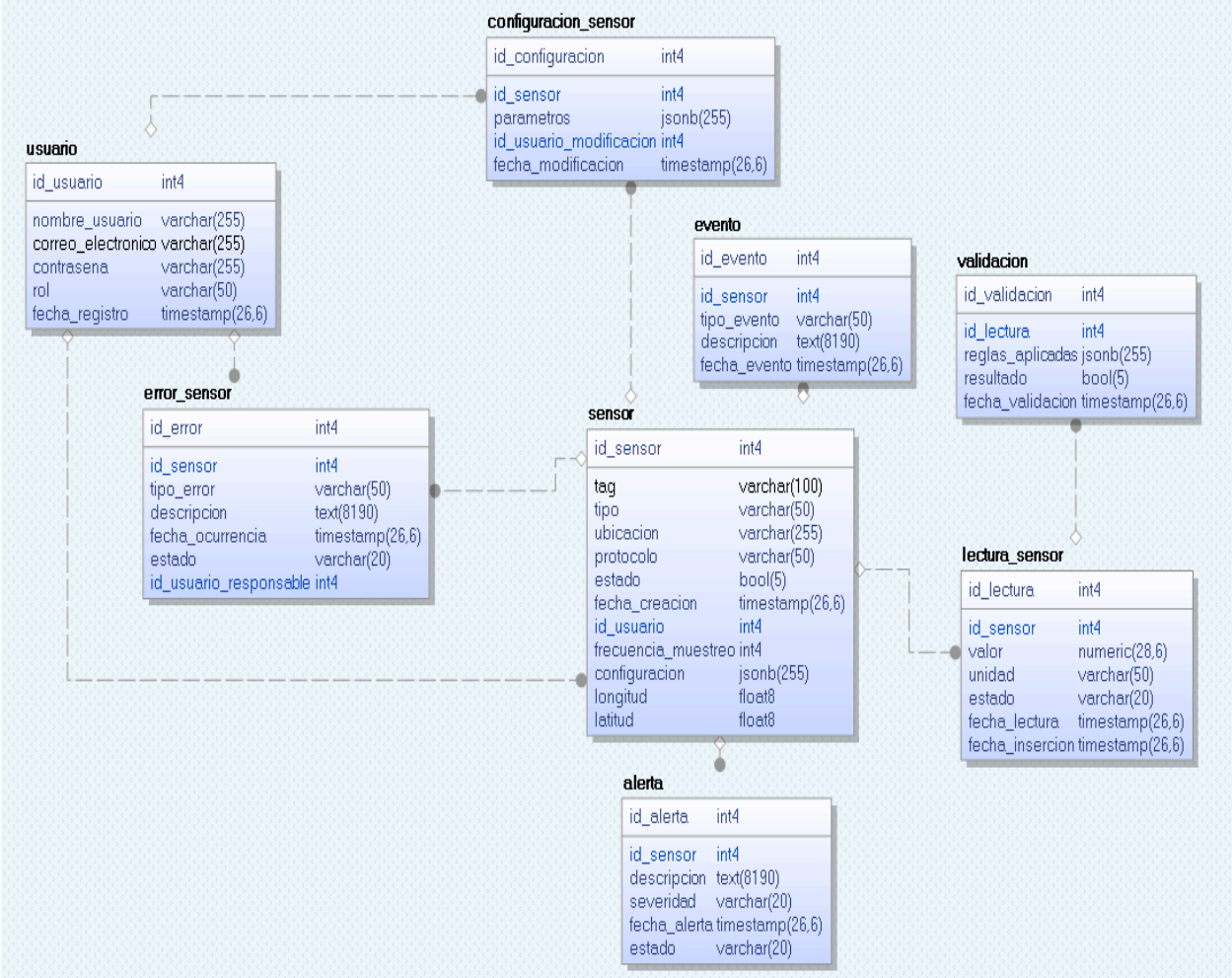
At Veolia, we are challenged to optimize the management and analysis of data coming from a broad range of sensors used in our water, waste, and energy management operations. To overcome this hurdle, we seek to develop a comprehensive platform that not only connects and manages these on-site sensors but also efficiently handles real-time information transmission through various communication protocols.

The goal is to create a robust solution that validates data quality and facilitates its seamless integration into our Data Lake on Google Cloud Platform (GCP). This platform should include an intuitive and accessible frontend for users, as well as a powerful backend to ensure efficiency and security in data processing and storage.

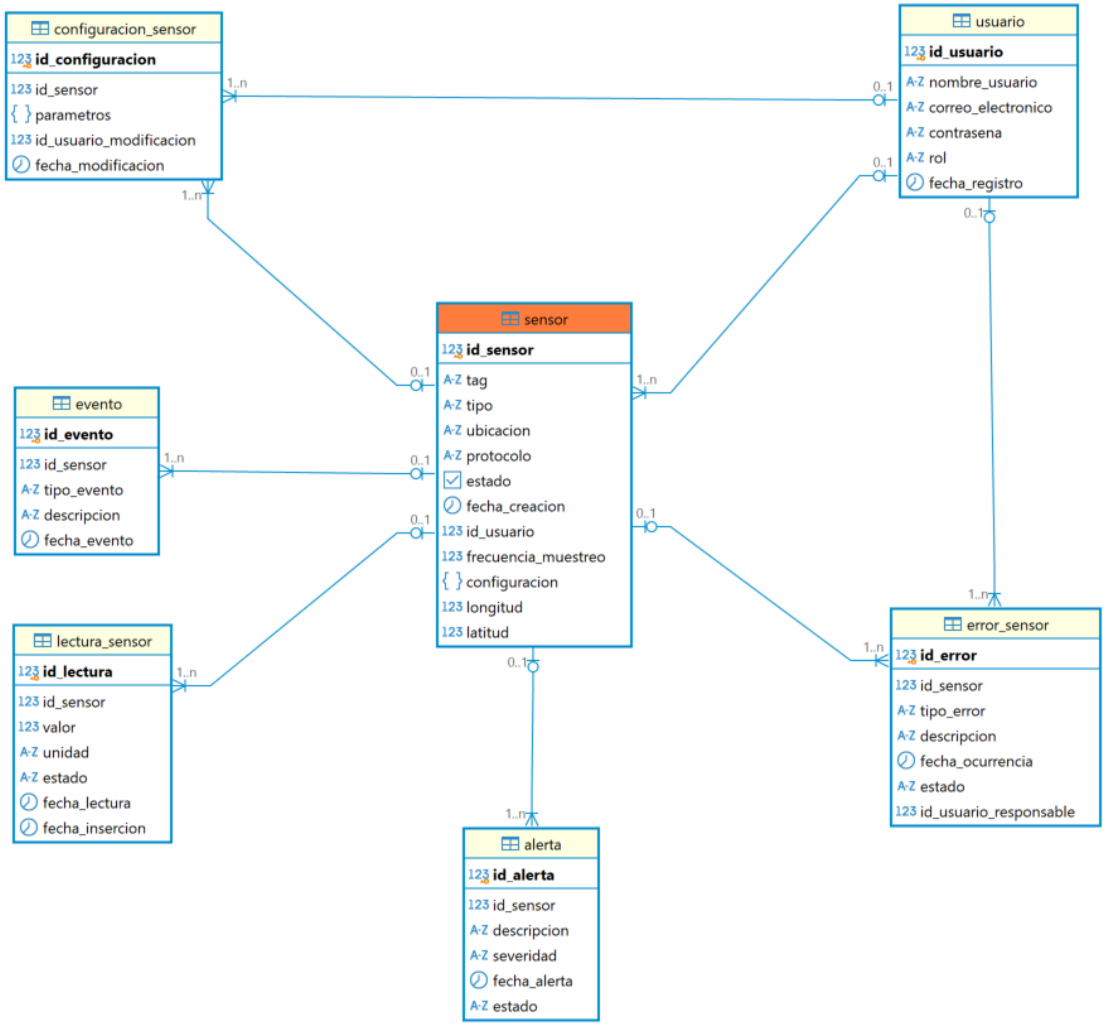
Moreover, the solution must be scalable and adaptable, capable of incorporating new sensors and data sources as our needs and technologies evolve.

# BackEnd

## Modelado en Erwin DM



## Modelado en DBeaver




## Consultas simples

| | |
|---|---|
| **Consulta de los eventos tipo de evento, sensor, fecha** | |
| **Consulta de alertas en sensores, id, tipo** | |
| **Consulta para listar todos los sensores activos** | `SELECT id_sensor, tag, tipo, ubicacion, protocolo` |

| | |
|---|---|
| Esta consulta proporciona una lista de todos los sensores que están actualmente activos en la plataforma. | `FROM sensor`<br>`WHERE estado = TRUE;` |
| **Consulta para contar el número total de sensores por tipo**<br>Esta consulta proporciona un resumen del número total de sensores, agrupados por tipo. | `SELECT tipo, COUNT(*) AS total_sensores`<br>`FROM sensor`<br>`GROUP BY tipo;` |
| **Consulta para obtener las últimas 10 lecturas de un sensor específico**<br>Esta consulta devuelve una lista de sensores que han registrado errores, incluyendo el tipo de error y la fecha de ocurrenca. | `SELECT id_lectura, valor, unidad, estado, fecha_lectura`<br>`FROM lectura_sensor`<br>`WHERE id_sensor = 1`<br>`ORDER BY fecha_lectura DESC`<br>`LIMIT 10;` |
| **Consulta para obtener los sensores que han tenido errores**<br>Esta consulta devuelve una lista de sensores que han registrado errores, incluyendo el tipo de error y la fecha de ocurrencia. | `SELECT s.id_sensor, s.tag, e.tipo_error, e.descripcion, e.fecha_ocurrencia`<br>`FROM sensor s`<br>`JOIN error_sensor e ON s.id_sensor = e.id_sensor;` |
| **Consulta para contar el número de alertas por severidad**<br>Esta consulta proporciona una vista general del número de alertas registradas en el sistema, agrupadas por su nivel de severidad. | `SELECT severidad, COUNT(*) AS total_alertas`<br>`FROM alerta`<br>`GROUP BY severidad;` |
| **Consulta para obtener todas las configuraciones de un sensor**<br>Esto muestra todas las configuraciones actuales para un sensor específico. | `SELECT id_configuracion, parametros, fecha_modificacion`<br>`FROM configuracion_sensor`<br>`WHERE id_sensor = 1`<br>`ORDER BY fecha_modificacion DESC;` |
| **Consulta para obtener el total de sensores administrados por cada usuario**<br>Esta consulta es útil para saber qué usuarios están gestionando la mayor cantidad de sensores | `SELECT u.nombre_usuario, COUNT(s.id_sensor) AS total_sensores`<br>`FROM usuario u`<br>`LEFT JOIN sensor s ON u.id_usuario = s.id_usuario`<br>`GROUP BY u.id_usuario;` |

# Consultas intermedias

| | |
|---|---|
| **Consulta de errores más frecuentes por tipo de sensor:**<br>Esta consulta muestra los tipos de errores más frecuentes para cada tipo de sensor. Es útil para identificar problemas comunes y priorizar soluciones | `SELECT`<br>`    s.tipo AS tipo_sensor,`<br>`    e.tipo_error,`<br>`    COUNT(e.id_error) AS frecuencia_errores`<br>`FROM`<br>`    error_sensor e`<br>`JOIN`<br>`    sensor s ON e.id_sensor = s.id_sensor`<br>`GROUP BY`<br>`    s.tipo, e.tipo_error`<br>`ORDER BY`<br>`    s.tipo, frecuencia_errores DESC;` |
| **Consulta de sensores con más alertas críticas en los últimos 30 días:**<br>Esta consulta identifica los sensores que han generado más alertas críticas en el último mes. Esto ayuda a detectar sensores problemáticos o situaciones críticas recurrentes. | `SELECT`<br>`    s.tag,`<br>`    COUNT(a.id_alerta) AS total_alertas_criticas`<br>`FROM`<br>`    alerta a`<br>`JOIN`<br>`    sensor s ON a.id_sensor = s.id_sensor`<br>`WHERE`<br>`    a.severidad = 'crítico'`<br>`    AND a.fecha_alerta >= NOW() - INTERVAL '30 days'`<br>`GROUP BY`<br>`    s.tag`<br>`ORDER BY`<br>`    total_alertas_criticas DESC;` |
| **Consulta de sensores fuera de los parámetros operativos:**<br>Esta consulta muestra los sensores que tienen lecturas fuera de los parámetros operativos definidos (por ejemplo, sensores que tienen lecturas fuera de un rango preestablecido). | `SELECT`<br>`    s.tag,`<br>`    l.valor,`<br>`    l.unidad,`<br>`    l.fecha_lectura`<br>`FROM`<br>`    lectura_sensor l`<br>`JOIN`<br>`    sensor s ON l.id_sensor = s.id_sensor`<br>`WHERE`<br>`    (l.valor < 10 OR l.valor > 100) -- Ejemplo de rango fuera de los parámetros`<br>`ORDER BY`<br>`    l.fecha_lectura DESC;` |

# Consultas avanzadas

| | |
|---|---|
| **Identificación de sensores problemáticos con frecuencia de errores**<br>Esta consulta identifica los sensores que han tenido la mayor cantidad de errores en el último año, agrupándolos por tipo de error y mostrando el usuario responsable de la mayoría de las resoluciones. | `WITH errores_por_sensor AS (`<br>`  SELECT`<br>`  s.id_sensor,`<br>`  s.tag,`<br>`  COUNT(e.id_error) AS total_errores,`<br>`  e.tipo_error,`<br>`  u.nombre_usuario AS usuario_responsable`<br>`  FROM`<br>`  error_sensor e`<br>`  JOIN`<br>`  sensor s ON e.id_sensor = s.id_sensor`<br>`  JOIN`<br>`  usuario u ON e.id_usuario_responsable = u.id_usuario`<br>`  WHERE`<br>`  e.fecha_ocurrencia >= NOW() - INTERVAL '1 year'`<br>`  GROUP BY`<br>`  s.id_sensor, s.tag, e.tipo_error, u.nombre_usuario`<br>`)`<br>`SELECT`<br>`  tag,`<br>`  tipo_error,`<br>`  MAX(total_errores) AS max_errores,`<br>`  usuario_responsable`<br>`FROM`<br>`  errores_por_sensor`<br>`GROUP BY`<br>`  tag, tipo_error, usuario_responsable`<br>`ORDER BY`<br>`  max_errores DESC;` |
| **Detección de sensores con alta variabilidad en el Tiempo**<br>Identificar sensores que muestran una alta variabilidad en sus lecturas a lo largo del tiempo, lo cual podría indicar problemas de calibración, fallas intermitentes, o condiciones operativas inestables. | `WITH sensor_variability AS (`<br>`  SELECT`<br>`  id_sensor,`<br>`  AVG(ABS(valor - AVG(valor) OVER (PARTITION BY id_sensor))) AS avg_variability`<br>`  FROM` |

| | |
|---|---|
| | lectura_sensor<br>WHERE<br>fecha_lectura >= NOW() - INTERVAL '60 days' -- Observa los últimos 60 días para tener una visión más amplia<br>GROUP BY<br>id_sensor, valor<br>)<br>SELECT<br>s.id_sensor,<br>s.tag,<br>sv.avg_variability<br>FROM<br>sensor s<br>JOIN<br>sensor_variability sv ON s.id_sensor = sv.id_sensor<br>WHERE<br>sv.avg_variability > (SELECT AVG(avg_variability) FROM sensor_variability) -- Sensores con variabilidad superior al promedio<br>ORDER BY<br>sv.avg_variability DESC; |
| **Análisis de causas raíz de fallos en sensores:**<br>Esta consulta identifica las causas raíz más comunes de fallos en los sensores. Ayuda a la empresa a entender por qué ciertos sensores fallan con frecuencia, lo que podría llevar a decisiones sobre la mejora del mantenimiento, reemplazo de equipos o ajustes en la configuración. | SELECT<br>tipo_error,<br>COUNT(id_error) AS total_fallos,<br>ROUND(AVG(EXTRACT(EPOCH FROM (CURRENT_TIMESTAMP - fecha_ocurrencia)) / 3600), 2) AS tiempo_promedio_desde_ultimo_fallo_horas<br>FROM<br>error_sensor<br>GROUP BY<br>tipo_error<br>ORDER BY<br>total_fallos DESC, tiempo_promedio_desde_ultimo_fallo_horas ASC; |
| **Sensores que requieren calibración frecuente:**<br><br>Identificar sensores que tienen que ser calibrados frecuentemente puede indicar problemas con los sensores, su configuración, o el entorno operativo. Esto permite mejorar la planificación de | SELECT<br>s.tag,<br>COUNT(e.id_evento) AS numero_calibraciones<br>FROM<br>evento e<br>JOIN<br>sensor s ON e.id_sensor = |

| | |
|---|---|
| mantenimiento. | s.id_sensor<br>WHERE<br>    e.tipo_evento = 'Calibración'<br>    AND e.fecha_evento >= NOW() -<br>INTERVAL '1 year'<br>GROUP BY<br>    s.tag<br>ORDER BY<br>    numero_calibraciones DESC; |
| **Consulta de sensores dentro de un radio específico de una ubicación:**<br><br>Esta consulta encuentra todos los sensores ubicados dentro de un radio determinado desde una ubicación geográfica específica. Esto puede ser útil para analizar sensores en una zona afectada por un evento o para organizar el mantenimiento. | -- Parámetros de ejemplo: Coordenadas del punto central (latitud y longitud) y el radio en kilómetros<br>WITH ubicacion_central AS (<br>    SELECT<br>        4.710989 AS lat_centro, --<br>Ejemplo de latitud (Bogotá)<br>        -74.072092 AS lon_centro, --<br>Ejemplo de longitud (Bogotá)<br>        10 AS radio -- Radio en<br>kilómetros<br>)<br>SELECT<br>    s.id_sensor,<br>    s.tag,<br>    s.ubicacion,<br>    s.latitud,<br>    s.longitud,<br>    (6371 *<br>acos(cos(radians(u.lat_centro)) *<br>cos(radians(s.latitud)) *<br>cos(radians(s.longitud) -<br>radians(u.lon_centro)) +<br>sin(radians(u.lat_centro)) *<br>sin(radians(s.latitud)))) AS<br>distancia_km<br>FROM<br>    sensor s, ubicacion_central u<br>WHERE<br>    (6371 *<br>acos(cos(radians(u.lat_centro)) *<br>cos(radians(s.latitud)) *<br>cos(radians(s.longitud) -<br>radians(u.lon_centro)) +<br>sin(radians(u.lat_centro)) *<br>sin(radians(s.latitud)))) <= u.radio<br>ORDER BY<br>    distancia_km; |

| | |
|---|---|
| **Consulta de sensores por región Geográfica:**<br>Esta consulta muestra todos los sensores ubicados en una región geográfica específica, definida por un cuadro delimitador (bounding box). Esto es útil para filtrar sensores por áreas operativas específicas. | `-- Parámetros de ejemplo: Coordenadas del cuadro delimitador`<br>`SELECT`<br>`  s.id_sensor,`<br>`  s.tag,`<br>`  s.ubicacion,`<br>`  s.latitud,`<br>`  s.longitud`<br>`FROM`<br>`  sensor s`<br>`WHERE`<br>`  s.latitud BETWEEN 4.5 AND 5.5 -- Latitud mínima y máxima de la región`<br>`  AND s.longitud BETWEEN -75.0 AND -73.5 -- Longitud mínima y máxima de la región`<br>`ORDER BY`<br>`  s.id_sensor;` |
| **Consulta de sensores cercanos al punto de mantenimiento programado**<br><br>Identifica sensores que están cerca de un punto de mantenimiento programado. Útil para optimizar rutas de mantenimiento y agrupar tareas de mantenimiento en la misma área geográfica. | `WITH puntos_mantenimiento AS (`<br>`  SELECT`<br>`  10.96854 AS lat_mantenimiento, -- Ejemplo de latitud (Cartagena)`<br>`  -74.78132 AS lon_mantenimiento, -- Ejemplo de longitud (Cartagena)`<br>`  5 AS radio -- Radio en kilómetros`<br>`)`<br>`SELECT`<br>`  s.id_sensor,`<br>`  s.tag,`<br>`  s.ubicacion,`<br>`  s.latitud,`<br>`  s.longitud,`<br>`  (6371 * acos(cos(radians(p.lat_mantenimiento)) * cos(radians(s.latitud)) * cos(radians(s.longitud) - radians(p.lon_mantenimiento)) + sin(radians(p.lat_mantenimiento)) * sin(radians(s.latitud)))) AS distancia_km`<br>`FROM`<br>`  sensor s, puntos_mantenimiento p`<br>`WHERE`<br>`  (6371 * acos(cos(radians(p.lat_mantenimiento)) * cos(radians(s.latitud)) * cos(radians(s.longitud) - radians(p.lon_mantenimiento)) + sin(radians(p.lat_mantenimiento)) * sin(radians(s.latitud)))) <= p.radio`<br>`ORDER BY`<br>`  distancia_km;` |